FIG.1 (PRIOR ART)

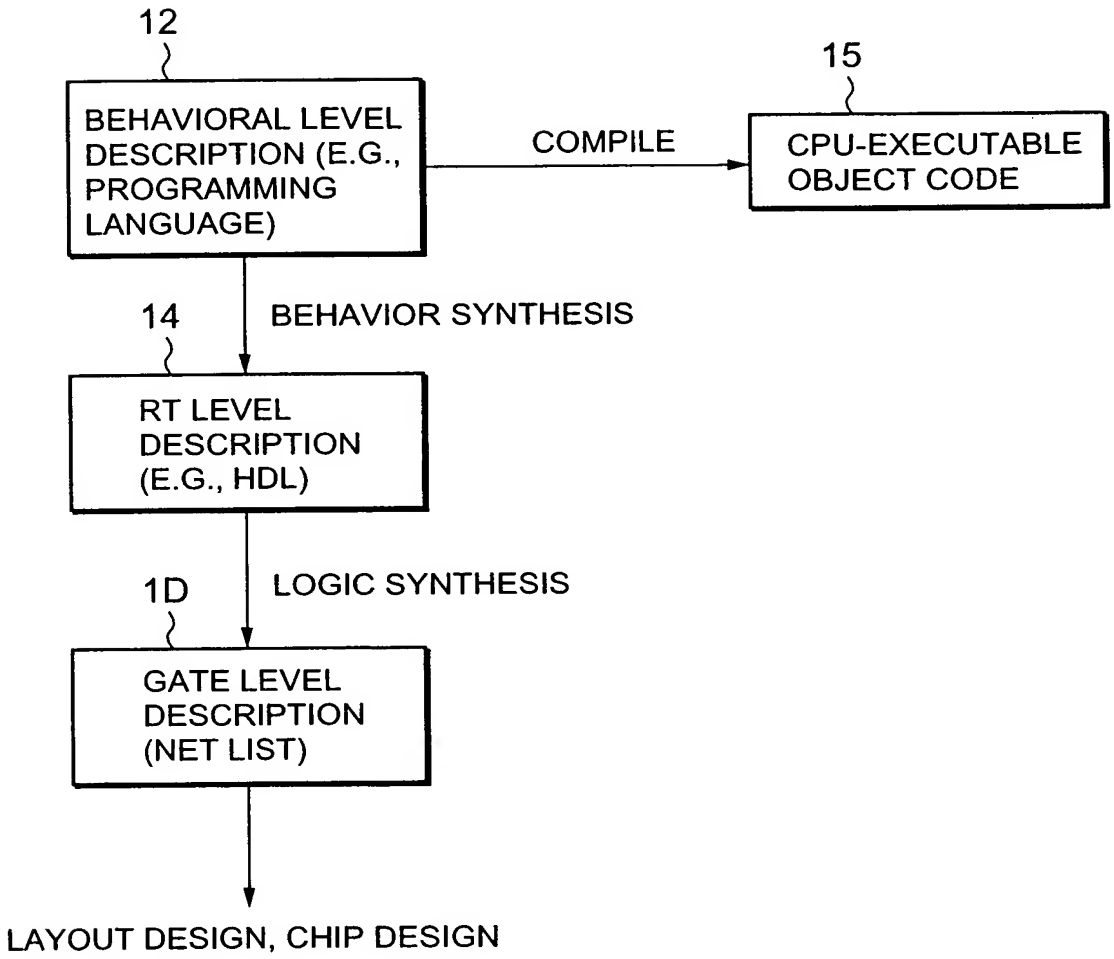


FIG.2 (PRIOR ART)

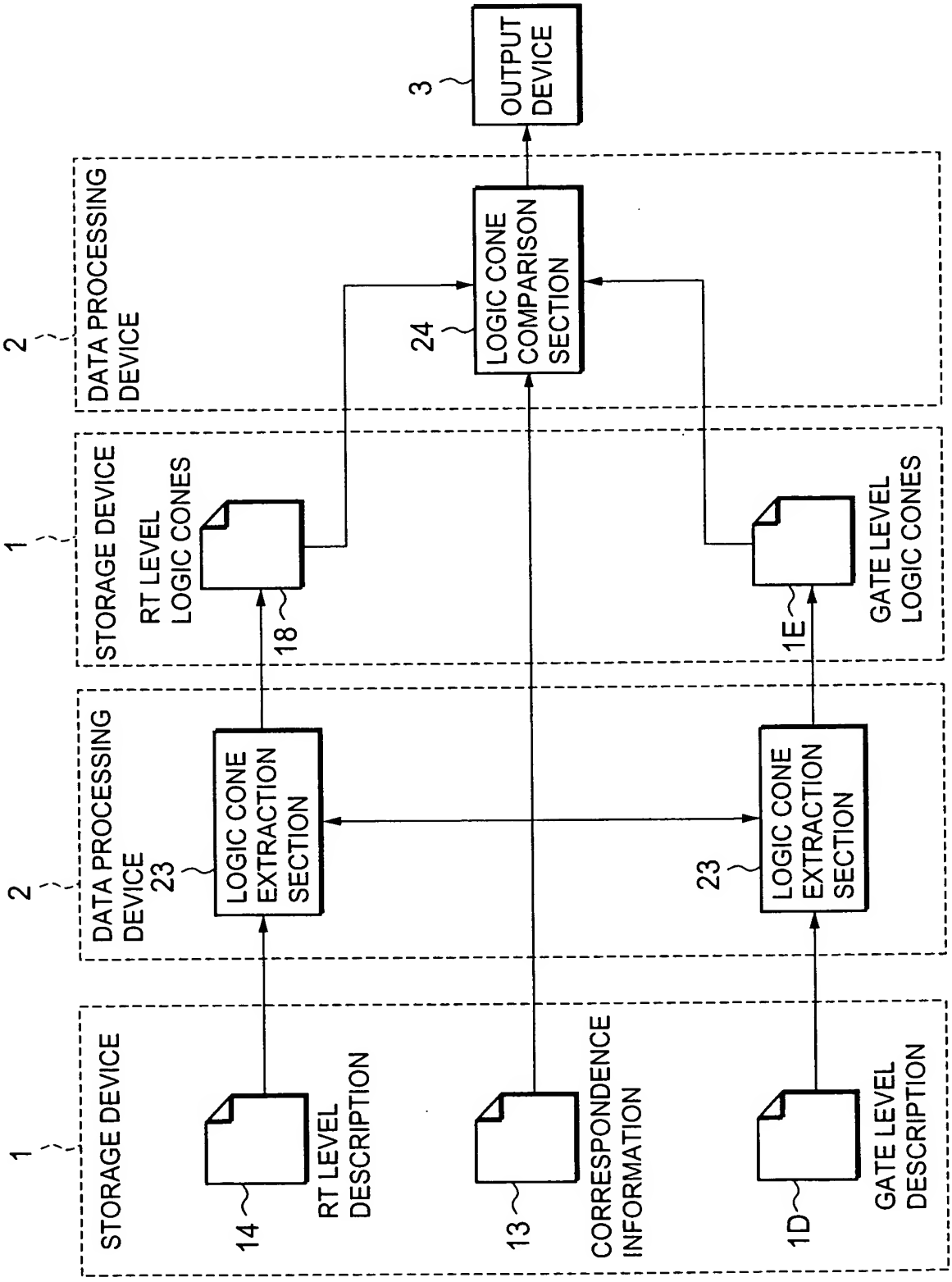


FIG.3 (PRIOR ART)

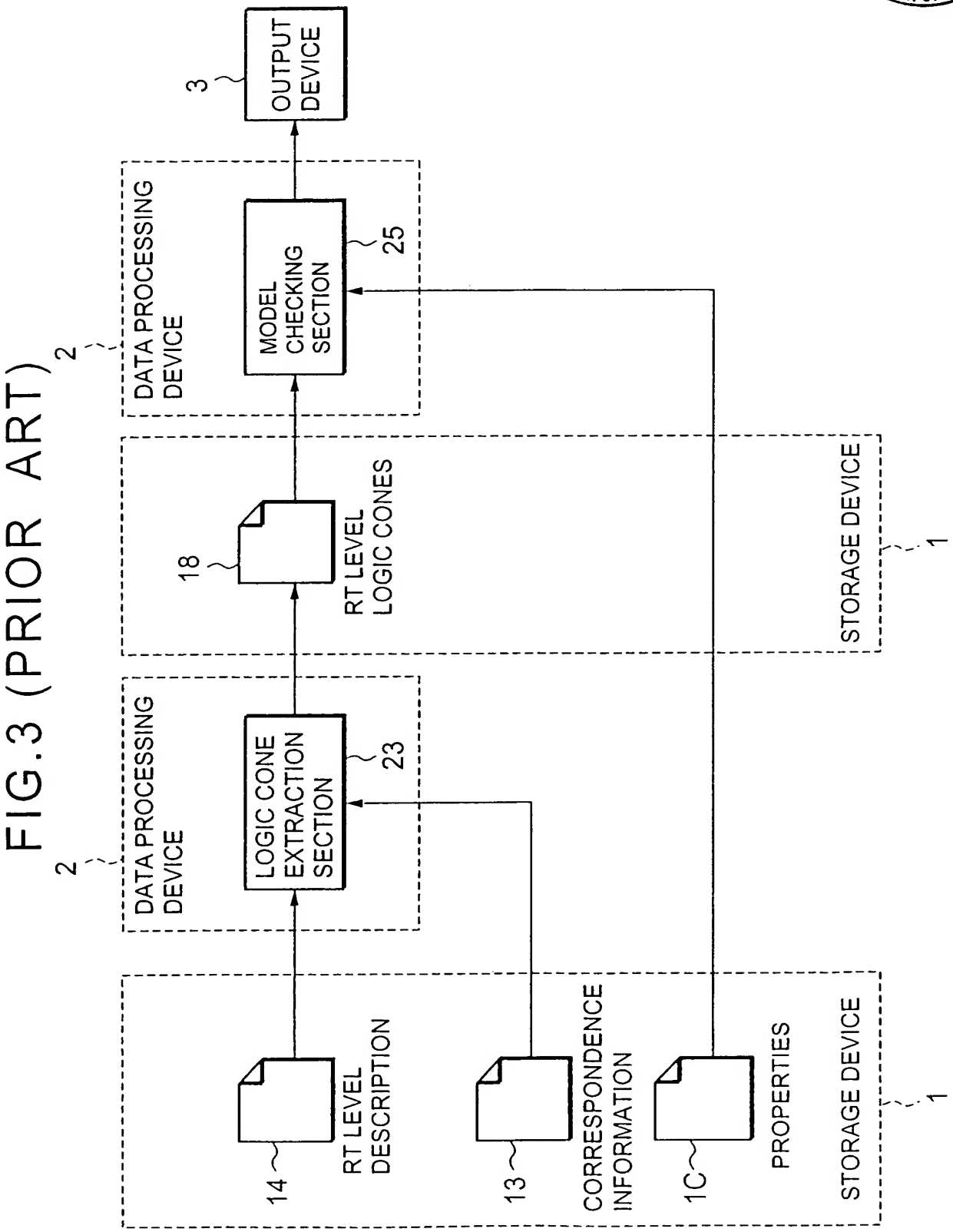


FIG. 4

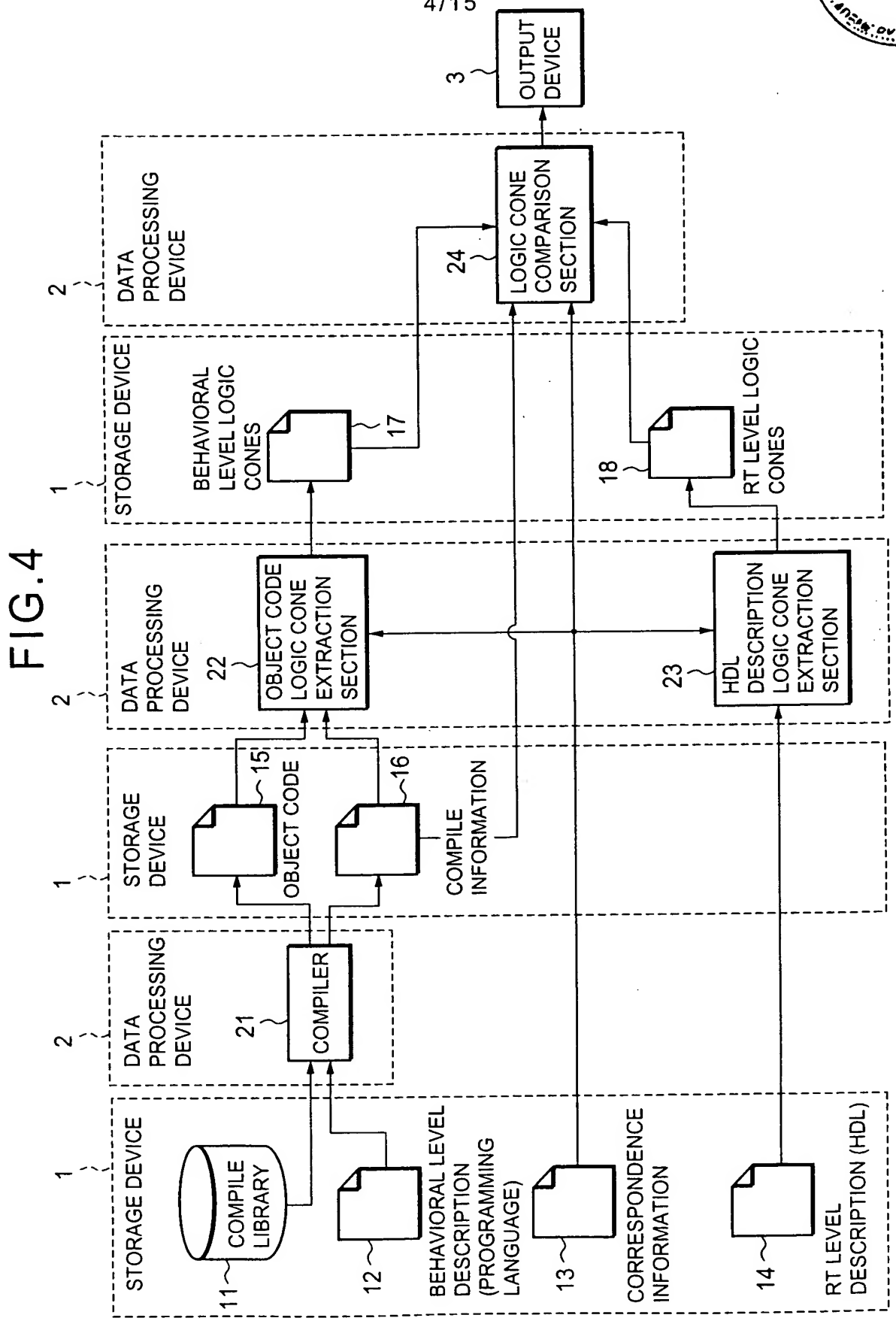


FIG.5

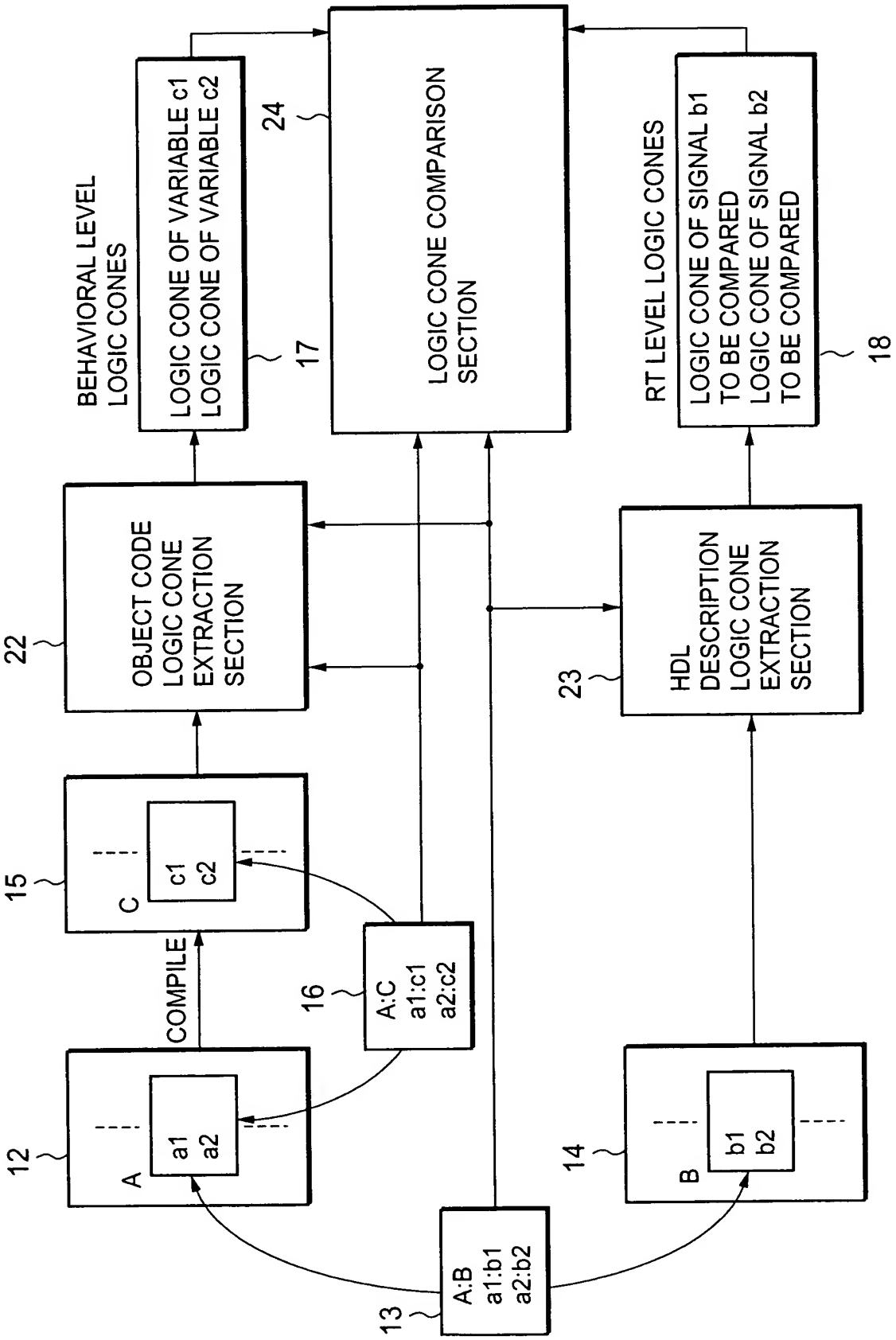


FIG.6

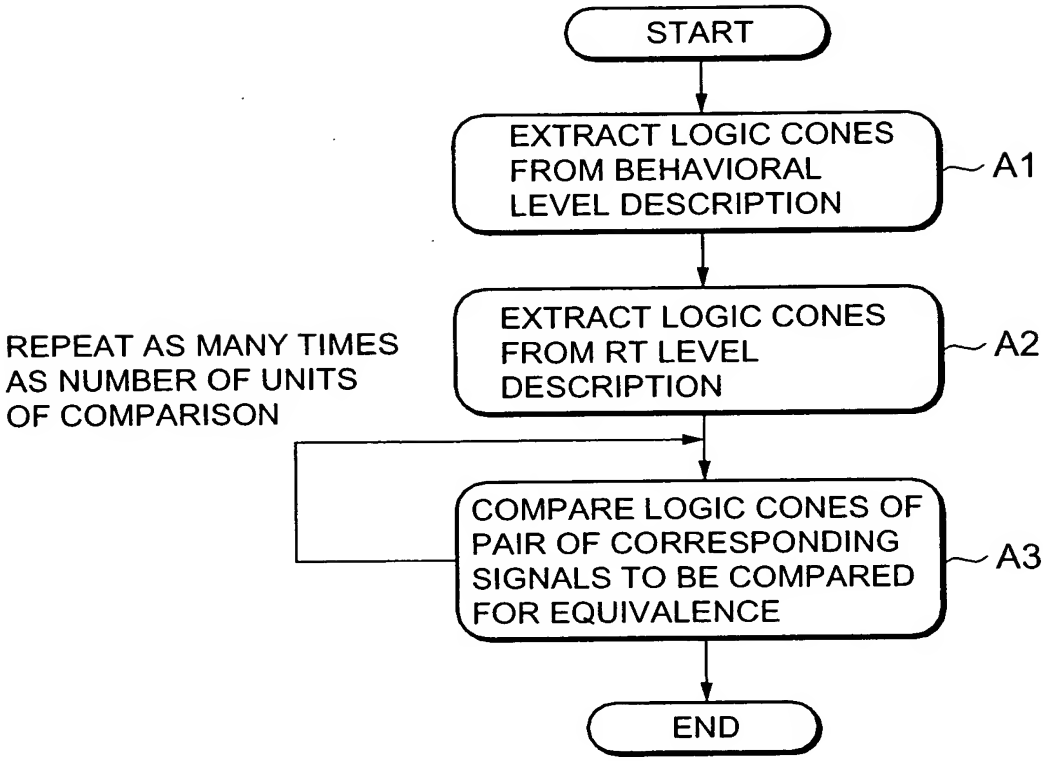


FIG.7

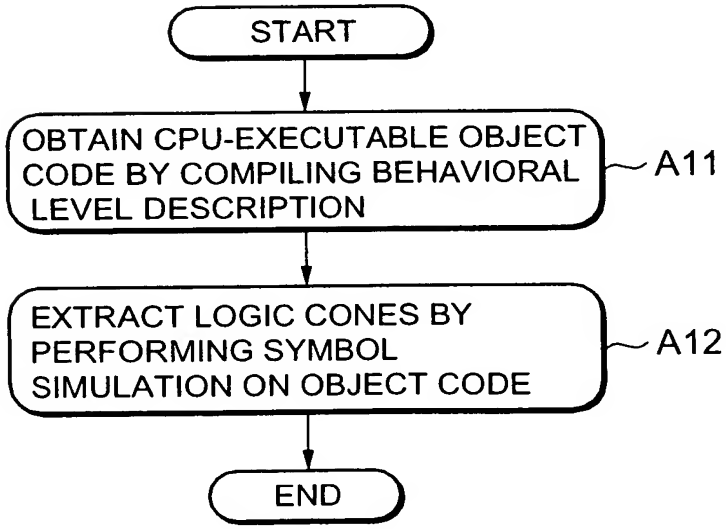


FIG.8

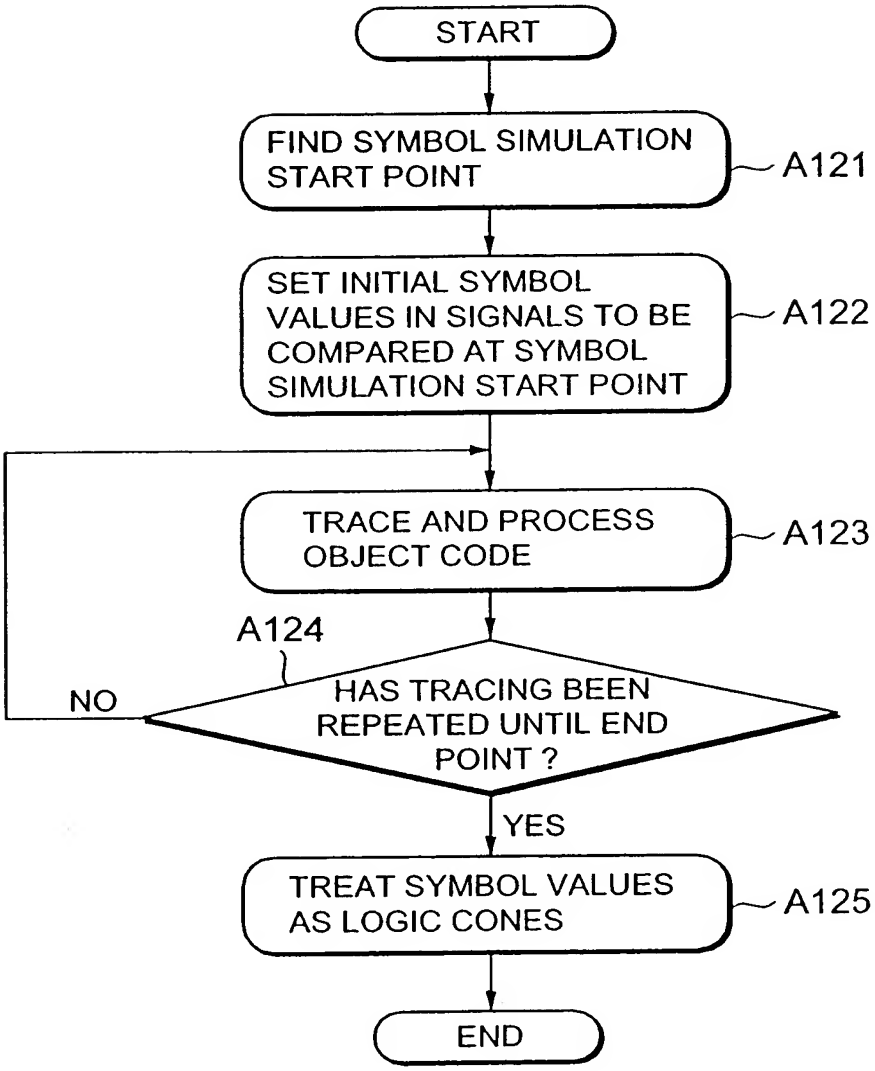


FIG.9

12

```
int a, b, c ;                          ·····(1)
int in0, out0 ;                        ·····(2)
#ifdef c                               ·····(3)
main ( ) {                             ·····(4)
  while (1) {                          ·····(5)
     scanf ("%d", &in0) ;              ·····(6)
     addition ( ) ;                    ·····(7)
     printf ("%d\n", out0) ;           ·····(8)
}                                      ·····(9)
#endif                                 ·····(10)

/* to behavioral Synthesis */         ·····(11)
void addition ( )                      ·····(12)
{                                      ·····(13)
    b = a + b ;                        ·····(14)
    a = in0 ;                          ·····(15)
    out0 = b ;                         ·····(16)
    return ;                           ·····(17)
}                                      ·····(18)
```

FIG.10

14

```
module addition (in0, out0, CLOCK) ;   ·····(1)
input [31:0] in0 ;                     ·····(2)
output [31:0] out0 ;                   ·····(3)
input CLOCK                            ·····(4)
reg [31:0] RG01 ;                      ·····(5)
reg [31:0] RG02 ;                      ·····(6)

assign out0 = RG02 ;                   ·····(7)

always @ ( posedge CLOCK)              ·····(8)
begin                                  ·····(9)
      RG01 <= in0 ;                    ·····(10)
      RG02 <= RG01 + RG02 ;            ·····(11)
end                                    ·····(12)
endmodule                              ·····(13)
```

FIG.11

15

```
addition :                          ·····(1)
    movl a, %eax                   ·····(2)
    addl %eax, b                   ·····(3)
    movl in0, %eax                 ·····(4)
    movl %eax, a                   ·····(5)
    movl b, %eax                   ·····(6)
    movl %eax, out0                ·····(7)
```

FIG.12

EXAMPLE OF CORRESPONDENCE INFORMATION (PARTIAL) 13

SIGNALS IN C DESCRIPTION	SIGNALS IN HDL DESCRIPTION
in0	in0
out0	out0
a	RG01
b	RG02

FIG.13

EXAMPLE OF COMPILE INFORMATION (PARTIAL)

SIGNALS IN C-LANGUAGE DESCRIPTION	STORAGE AREA IN OBJECT CODE
in0	in0
out0	out0
a	a
b	b

FIG.14

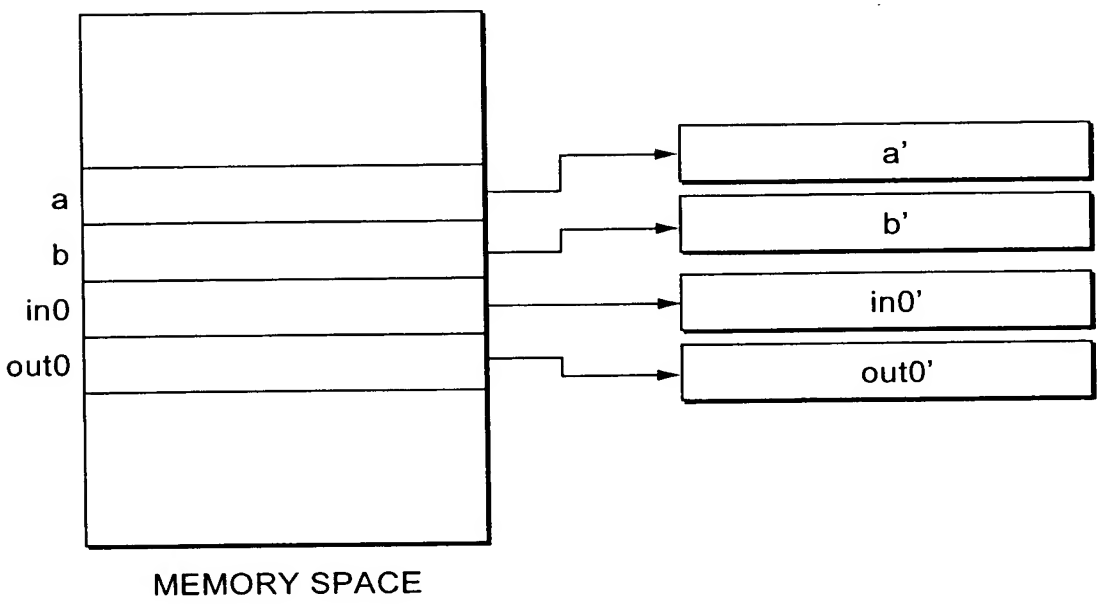


FIG.15

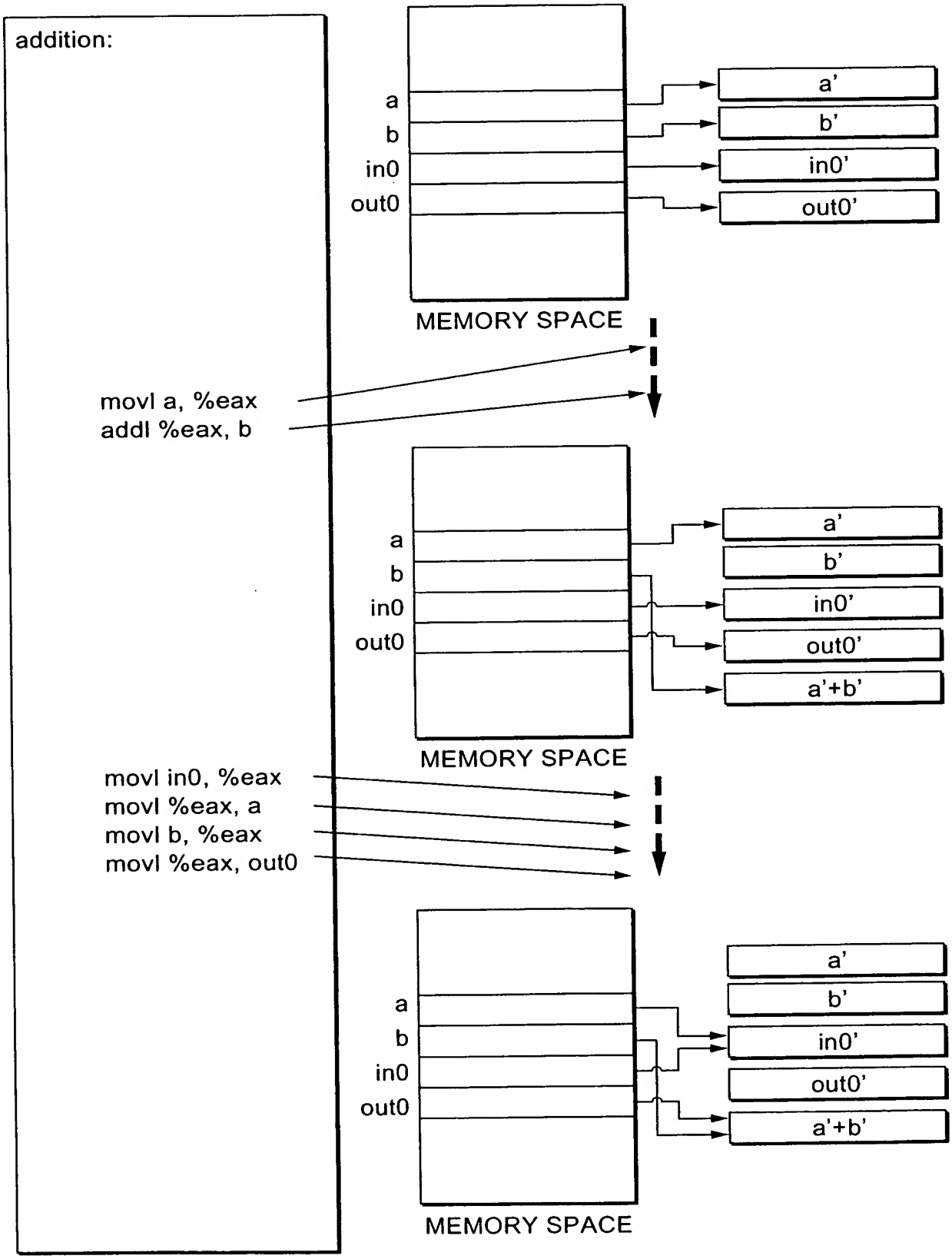


FIG.16

EXAMPLES OF LOGIC CONES IN PROGRAM DESCRIPTION

VARIABLE	LOGIC CONE
a	in0'
b	a'+b'
in0	in0'
out0	a'+b'

FIG.17

EXAMPLES OF LOGIC CONES IN HDL DESCRIPTION

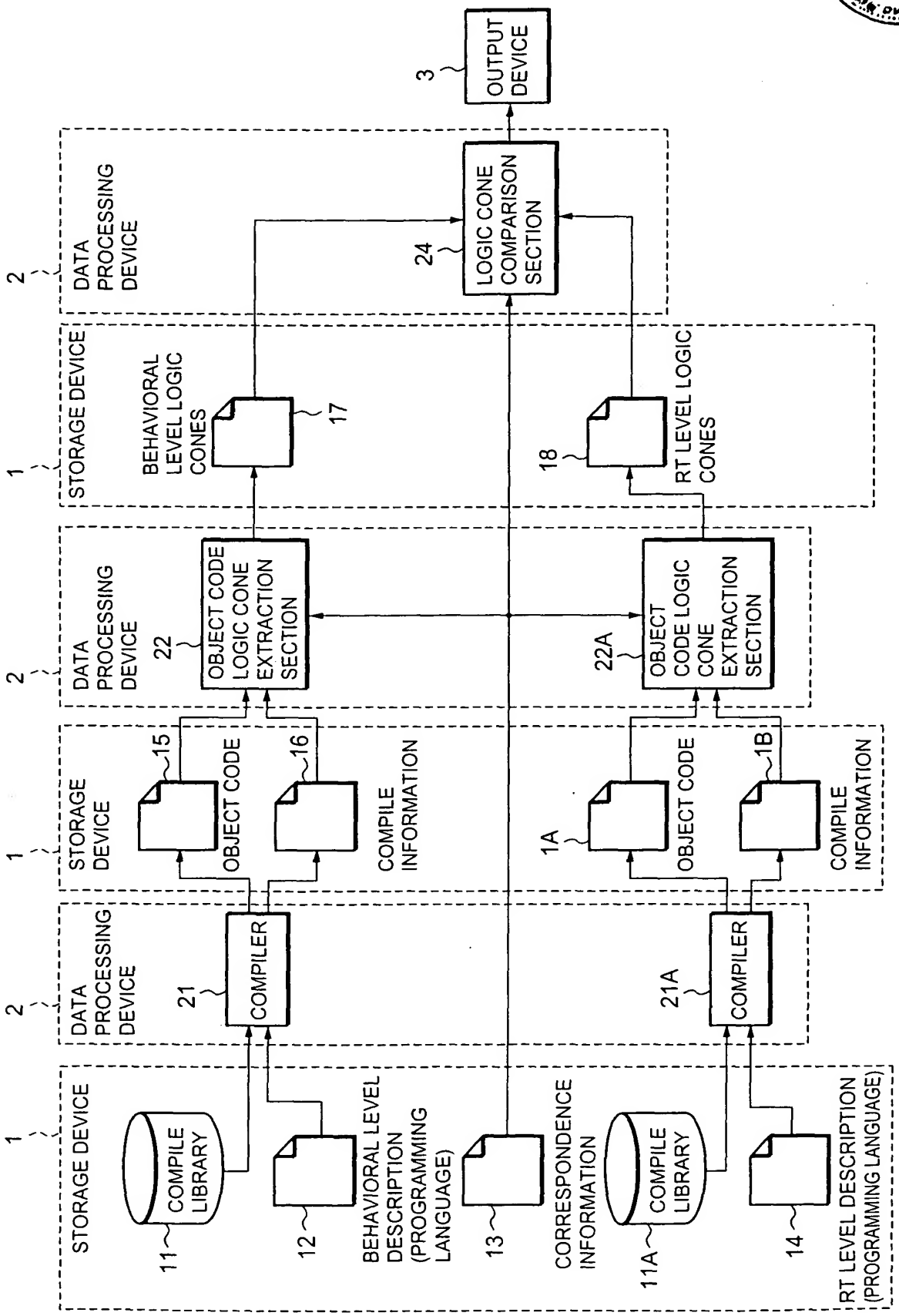
VARIABLE	LOGIC CONE
RG01	in0'
RG02	RG01'+RG02'
in0	in0'
out0	RG01'+RG02'

FIG.18

CORRESPONDENCE BETWEEN VARIABLES
IN OBJECT CODE AND SIGNALS IN HDL

VARIABLE IN OBJECT CODE	SIGNAL IN HDL
a	RG01
b	RG02
in0	in0
out0	out0

FIG.19



Title: LOGIC VERIFICATION AND LOGIC CONE EXTRACTION TECHNIQUE
Inventor(s): Takashi TAKENAKA
Appl. No.: 10/612,193

FIG.20

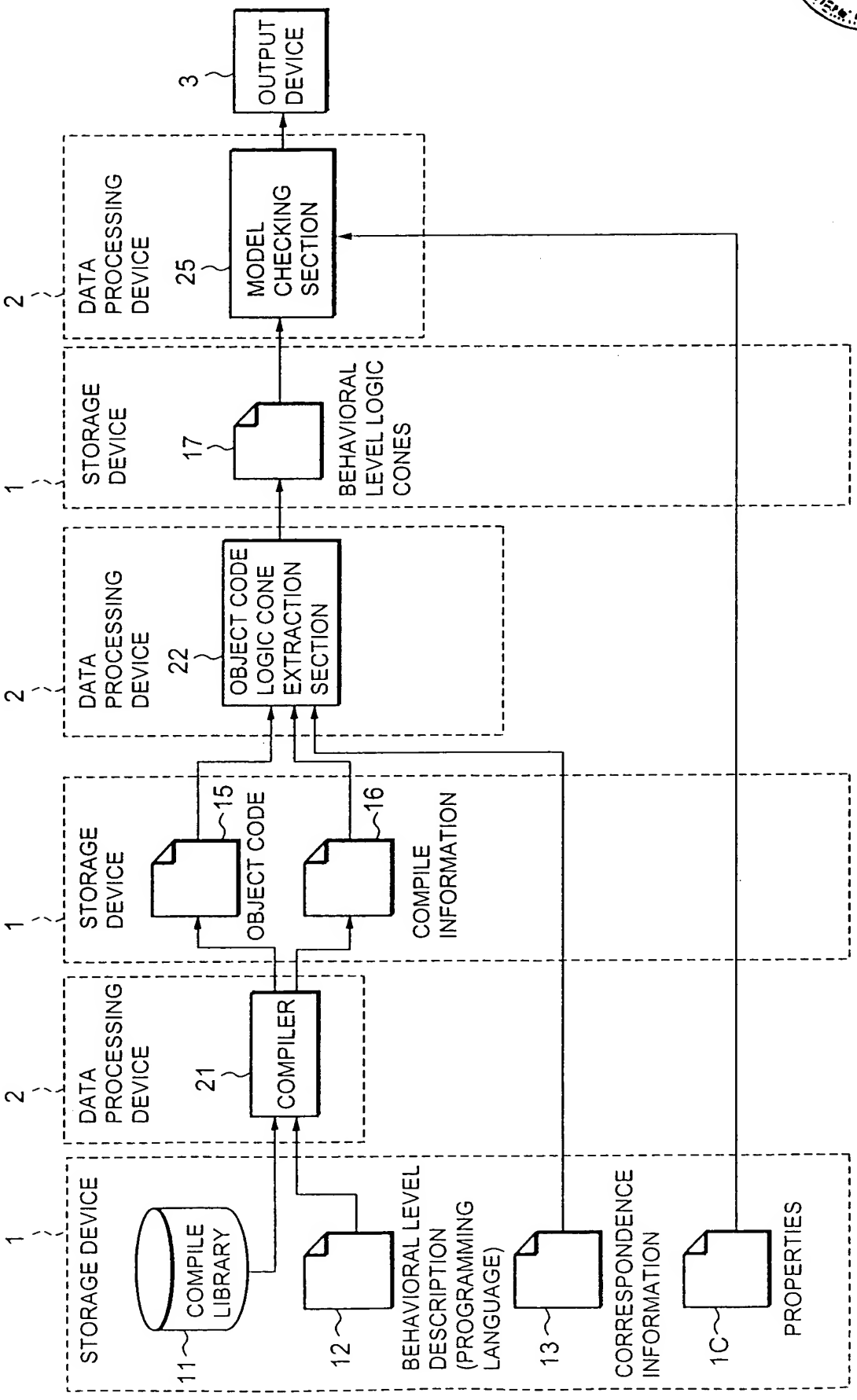


FIG.21